FP.1 Match 3D Objects

matchBoundingBox function is implemented in camFusion_Student.cpp.

FP.2 Compute Lidar-based TTC

computeTTCLidar function is implemented in camFusion_Student.cpp. Outliers are handled by taking the median x of the preceding car's lidar point cloud instead of taking the closest point.

FP.3 Associate Keypoint Correspondences with Bounding Boxes

clusterKptMatcheswithROI function is implemented in camFusion_Student.cpp. Calculated euclidean distances between matched keypoints, took the mean of all distances and excluded those points which had the distance above set minimum threshold distance value.

FP.4 Compute Camera-based TTC

ComputeTTCCamera is implemented camFusion_Student.cpp. Used median distance ratio to handle the outliers.

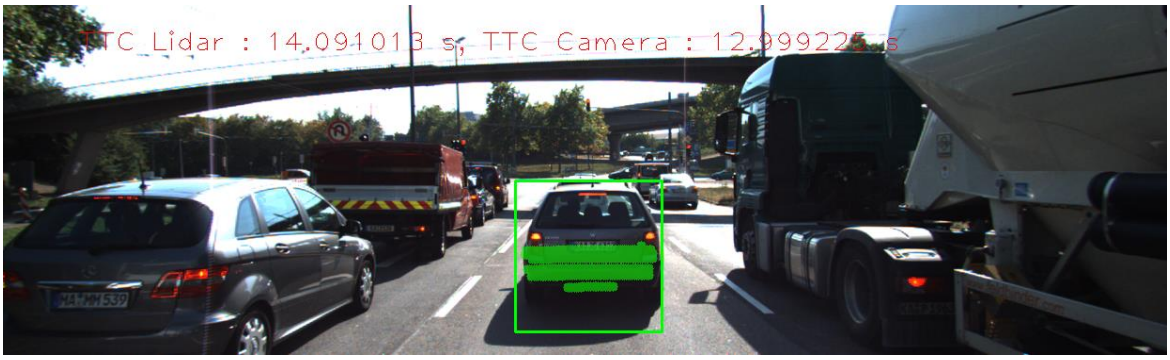
***All the results corresponding to the below performance evaluation are included inside the spreadsheet in the "results" folder.**

FP.5 Performance Evaluation 1

As can be seen in from the results, in between 2-3 and 3-4 ttc increased from 12.61 to 14.1 sec, and between 3-4 and 4-5 frame the ttc increased from 14.1 to 16.7 sec. This doesn't seem plausible. Possible reason could be the median criteria applied to avoid outliers might not be giving the distance of the exterior point of the preceding vehicle.

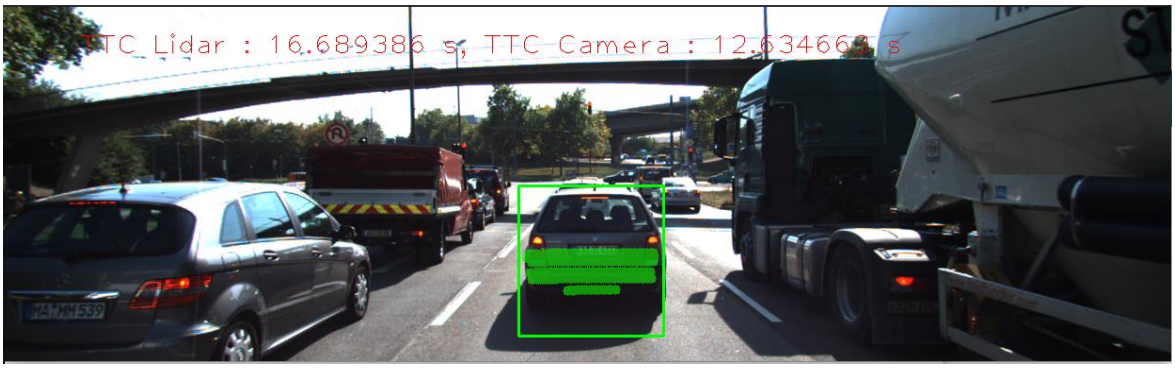
3-4

id=5, #pts=321
xmin=7.79 m, yw=1.47 m



4-5

id=6, #pts=319
xmin=7.68 m, yw=1.45 m



FP.6 Performance Evaluation 2

For TTC Camera:

SHITOMASI feature detector with all the feature descriptors except AKAZE gave good results. AKAZE feature detector also performed well. Whereas HARRIS, FAST and ORB gave ttc values as –inf or nan. Possible reason for the worst performance could be less number of keypoints detected onto the preceding vehicle.

For TTC Lidar:

It remained same for all the combinations of detector and descriptor per frame.